UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/713,887	11/16/2000	Paul L. Sinclair	8779	9423

26890 7590 07/17/2006

JAMES M. STOVER
NCR CORPORATION
1700 SOUTH PATTERSON BLVD, WHQ4
DAYTON, OH 45479

EXAMINER	
ALI, MOHAMMAD	

ART UNIT	PAPER NUMBER
2166	

DATE MAILED: 07/17/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/713,887	SINCLAIR ET AL.	
	Examiner	Art Unit	
	Mohammad Ali	2166	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 May 2006*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This is Non-Final office action to the pending claims 1-30.

In view of the appeal Brief filed on 05/03/06, prosecution is hereby reopened. A new rejection to pending claims 1-30 is set forth below.

To avoid abandonment of the application, appellant must exercise one of the following options:

(1) file a reply under 37 CFR 1.111 (if this office action is non-final) or a reply under 37 CFR 1.113 (if this office action is final); or,

(2) request reinstatement of the appeal.

If reinstatement of the appeal is requested, such request must be accompanied by a supplemental appeal brief, but no new amendments, affidavits (37 CFR 1.130, 1.131 or 1.132) or other evidence are permitted. See 37 CFR 1.193 (b)(2).

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claims 1-30 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

4. Regarding claims 1-30, the phrase "at some point" renders the claim indefinite because it is unclear whether the limitations following the phrase are part of the claimed invention; the phrase "at some point" renders the claim(s) indefinite because the

claim(s) include(s) elements not actually disclosed (those encompassed by "at some point"), thereby rendering the scope of the claim(s) unascertainable. See MPEP § 2173.05(d).

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

6. Claims 1-27 and 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Friske et al. ('Friske' hereinafter), US Patent 6,070,170 in view of Cejtin et al. ('Cejtin' hereinafter), USP, 5,745,703.

As to claim 1, Friske discloses a method for use in managing data in a database system (col. 2, 60-67). Friske teaches 'receiving a request to perform an operation on a set of target data residing in the database' as the unloaded target data set is reorganized by the processor 106 and loaded into a shadow location 310 of the storage unit 108 (col. 6, lines 25-27 et seq). Further, Friske teaches 'executing the operation in the database on the set of target data' as providing substantially continuous access to the database while reorganizing process is executing or waiting to execute. A data set is subject to reorganize in the target data set such as set of pages from the logical database (see col. 3, lines 26-27 and col. 6, lines 5-7). Finally, Friske teaches 'at some point after execution has begun, placing a lock on the target data to prevent concurrent execution of other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

Friske does not explicitly indicate claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data.

Cejtin discloses claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data (If the target address space of a remote-run! operation is the current address space, a new thread is simply spawned to execute the application. Otherwise, a message is constructed and sent across the channel connecting the source and target address spaces. The send-message procedure is responsible for writing a message to the target address space. Besides the actual message, send-message requires knowing the type of message being sent; in this case, the type is "run". It also needs to know the appropriate output channel, and requires access to a lock to prevent other messages from being sent to the target by threads concurrently executing on its address space while the transfer is underway. Send-message linearizes the message and associated data and releases the lock when complete, see col. 10, lines 18-31, Cejtin).

It would have been obvious to one ordinary skill in the data processing art at the time of the present invention to combine the teachings of the cited references because placing a lock on the target data to prevent concurrent execution of other operations on the target data of Cejtin teaching would have allowed Friske's system to much progress in building type systems and optimizers for Scheme that catch many potential type errors statically in order to significantly alleviate debugging overheads that would otherwise be incurred as suggested by Cejtin at col. 5, lines 29-33.

As to claim 2, Friske teaches 'placing an initial lock on the target data at a level that prevents concurrent execution of at least one operation and, at some point after execution has begun, placing a final lock on the target data at a level that prevents concurrent execution of a larger set of operations' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 3, Friske teaches 'the initial lock allows concurrent execution of operations that involve reading the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task

408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to clam 4, Friske teaches 'the final lock prevents concurrent execution of all operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 5, Friske teaches 'allowing a user to specify the type of lock initially placed on the data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the

target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 6, Friske teaches 'the operation is one of the following types: a COLLECT STATISTICS operation, a CREATE INDEX operation, and an ALTER' as after the target data set has been unloaded, the data is ordered in logical sequence in task 410, reorganized in task 412, and loaded into a shadow location in task 414. The target data set may include data indexes which, after the target data set has been reorganized, may be rebuilt into reorganized data indexes in task 416. Rebuilding the data indexes is necessary in the preferred embodiment as discussed above so that quick access to the reorganized data may occur. Log records are applied to the target data set in the shadow location in task 418 which allows any changes to the original data set which occurred while the reorganization was taking place to be applied to the reorganized target data set (coll. 7, lines 55-67 et seq).

As to claim 7, Friske discloses a database system (col. 2, lines 60-67). Friske teaches 'at least one storage device' as one or more magnetic data storage disks such as a "hard drive" or any other suitable storage device (col. 4, lines 20-23 et seq). Further, Friske teaches 'at least one computing node configured to deliver data to and retrieve data from the storage device' as storage comprises, for example, one or more magnetic data storage disks such as a "hard drive" or any other suitable storage device. The client ('node') computer 102 may include in one embodiment an output module 112

for outputting/displaying program status results on a graphic display 116, print mechanism 114 or data storage medium 118 (col. 4, lines 20-26 et seq). Friske discloses a database-management component (col. 2, lines 60-67). Friske teaches 'receiving a request to perform an operation on a set of target data residing in the database' as the unloaded target data set is reorganized by the processor 106 and loaded into a shadow location 310 of the storage unit 108 (col. 6, lines 25-27 et seq). Further, Friske teaches 'executing the operation in the database on the set of target data' providing substantially continuous access to the database while reorganizing process is executing or waiting to execute. A data set is subject to reorganize in the target data set such as set of pages from the logical database (see col. 3, lines 26-27 and col. 6, lines 5-7). Finally, Friske teaches 'at some point after execution has begun, placing a lock on the target data to prevent concurrent execution of other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

Friske does not explicitly indicate claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data.

Cejtin discloses claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data (If the target address space of a remote-run! operation is the current address space, a new thread is simply spawned to execute the application. Otherwise, a message is constructed and sent across the channel connecting the source and target address spaces. The send-message procedure is responsible for writing a message to the target address space. Besides the actual message, send-message requires knowing the type of message being sent; in this case, the type is "run". It also needs to know the appropriate output channel, and requires access to a lock to prevent other messages from being sent to the target by threads concurrently executing on its address space while the transfer is underway. Send-message linearizes the message and associated data and releases the lock when complete, see col. 10, lines 18-31, Cejtin).

It would have been obvious to one ordinary skill in the data processing art at the time of the present invention to combine the teachings of the cited references because placing a lock on the target data to prevent concurrent execution of other operations on the target data of Cejtin teaching would have allowed Friske's system to much progress in building type systems and optimizers for Scheme that catch many potential type errors statically in order to significantly alleviate debugging overheads that would otherwise be incurred as suggested by Cejtin at col. 5, lines 29-33.

As to claim 8, Friske teaches 'the database-management system is configured to place an initial lock on the target data at a level that prevents concurrent execution of at least one operation and, at some point after execution has begun, placing a final lock on the target data at a level that prevents concurrent execution of a larger set of operations' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 9, Friske teaches 'the initial lock allows concurrent execution of at least one other operation on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task

408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 10, Friske teaches 'the subsequent lock prevents concurrent execution of all other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 11, Friske teaches 'the database-management system is configured to allow a user to specify the type of lock initially placed on the data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B

waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 12, Friske teaches 'multiple computing nodes and multiple storage devices, where each storage node is configured to manage storage of data on at least a subset of the storage devices' as one or more magnetic data storage disks such as a "hard drive" or any other suitable storage device (col. 4, lines 20-23 et seq).

As to claim 13, Friske teaches 'the database-management system is configured to place the lock on a block of data that is spread across more than one of the storage devices' as one or more magnetic data storage disks such as a "hard drive" or any other suitable storage device (col. 4, lines 20-23 et seq).

As to claim 14, Friske teaches 'the operation is one of the following types: a COLLECT STATISTICS operation, a CREATE INDEX operation, and an ALTER TABLE operation' as after the target data set has been unloaded, the data is ordered in logical sequence in task 410, reorganized in task 412, and loaded into a shadow location in task 414. The target data set may include data indexes which, after the target data set has been reorganized, may be rebuilt into reorganized data indexes in task 416. Rebuilding the data indexes is necessary in the preferred embodiment as discussed above so that quick access to the reorganized data may occur. Log records are applied

to the target data set in the shadow location in task 418 which allows any changes to the original data set which occurred while the reorganization was taking place to be applied to the reorganized target data set (col. 7, lines 55-67 et seq).

As to claim 15, Friske discloses a computer program, stored on at least one computer-readable storage medium, for use in managing data in a database system, comprising executable instructions that, when executed by a computer (col. 2, 60-67). Friske teaches 'receiving a request to perform an operation on a set of target data residing in the database' as the unloaded target data set is reorganized by the processor 106 and loaded into a shadow location 310 of the storage unit 108 (col. 6, lines 25-27 et seq). Further, Friske teaches 'executing the operation in the database on the set of target data' as providing substantially continuous access to the database while reorganizing process is executing or waiting to execute. A data set is subject to reorganize in the target data set such as set of pages from the logical database (see col. 3, lines 26-27 and col. 6, lines 5-7). Finally, Friske teaches 'at some point after execution has begun, placing a lock on the target data to prevent concurrent execution of other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would

have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

Friske does not explicitly indicate claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data.

Cejtin discloses claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data (If the target address space of a remote-run! operation is the current address space, a new thread is simply spawned to execute the application. Otherwise, a message is constructed and sent across the channel connecting the source and target address spaces. The send-message procedure is responsible for writing a message to the target address space. Besides the actual message, send-message requires knowing the type of message being sent; in this case, the type is "run". It also needs to know the appropriate output channel, and requires access to a lock to prevent other messages from being sent to the target by threads concurrently executing on its address space while the transfer is underway. Send-message linearizes the message and associated data and releases the lock when complete, see col. 10, lines 18-31, Cejtin).

It would have been obvious to one ordinary skill in the data processing art at the time of the present invention to combine the teachings of the cited references because placing a lock on the target data to prevent concurrent execution of other operations on the target data of Cejtin teaching would have allowed Friske's system to much progress in building type systems and optimizers for Scheme that catch many potential type

errors statically in order to significantly alleviate debugging overheads that would otherwise be incurred as suggested by Cejtin at col. 5, lines 29-33.

As to claim 16, Friske teaches 'the program causes the computer to place an initial lock on the target data at a level that prevents concurrent execution of at least one operation and, at some point after execution has begun, placing a final lock on the target data at a level that prevents concurrent execution of a larger set of operations' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13 et seq).

As to claim 17, Friske teaches 'the initial lock allows concurrent execution of at least one other operation on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting

for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 18, Friske teaches 'the subsequent lock prevents concurrent execution of all other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 19, Friske teaches 'the program causes the computer to allow a user to specify the type of lock initially placed on the data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A,

already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 20, Friske teaches 'the operation is one of the following types: a COLLECT STATISTICS operation, a CREATE INDEX operation, and an ALTER TABLE operation' as after the target data set has been unloaded, the data is ordered in logical sequence in task 410, reorganized in task 412, and loaded into a shadow location in task 414. The target data set may include data indexes which, after the target data set has been reorganized, may be rebuilt into reorganized data indexes in task 416. Rebuilding the data indexes is necessary in the preferred embodiment as discussed above so that quick access to the reorganized data may occur. Log records are applied to the target data set in the shadow location in task 418 which allows any changes to the original data set which occurred while the reorganization was taking place to be applied to the reorganized target data set (col. 7, lines 55-67 et seq).

As to claim 21, Friske teaches a method for use in managing data in a database system (col. 2, 60-67). Friske teaches 'receiving a request to perform a data-definition operation on a set of target data residing in the database' as the unloaded target data

set is reorganized by the processor 106 and loaded into a shadow location 310 of the storage unit 108 (col. 6, lines 25-27 et seq). Further, Friske teaches 'executing the operation in the database on the set of target data' as providing substantially continuous access to the database while reorganizing process is executing or waiting to execute. A data set is subject to reorganize in the target data set such as set of pages from the logical database (see col. 3, lines 26-27 and col. 6, lines 5-7). Finally, Friske teaches 'at some point after execution has begun, placing a lock on the target data to prevent concurrent execution of other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

Friske does not explicitly indicate claimed placing a lock on the target data to excludes concurrent execution of other operations on the target data.

Cejtin discloses claimed placing a lock on the target data to excludes concurrent execution of other operations on the target data (If the target address space of a

remote-run! operation is the current address space, a new thread is simply spawned to execute the application. Otherwise, a message is constructed and sent across the channel connecting the source and target address spaces. The send-message procedure is responsible for writing a message to the target address space. Besides the actual message, send-message requires knowing the type of message being sent; in this case, the type is "run". It also needs to know the appropriate output channel, and requires access to a lock to prevent other messages from being sent to the target by threads concurrently executing on its address space while the transfer is underway. Send-message linearizes the message and associated data and releases the lock when complete, see col. 10, lines 18-31, Cejtin).

It would have been obvious to one ordinary skill in the data processing art at the time of the present invention to combine the teachings of the cited references because placing a lock on the target data to excludes concurrent execution of other operations on the target data of Cejtin teaching would have allowed Friske's system to much progress in building type systems and optimizers for Scheme that catch many potential type errors statically in order to significantly alleviate debugging overheads that would otherwise be incurred as suggested by Cejtin at col. 5, lines 29-33.

As to claim 22, Friske teaches 'the initial lock excludes at least some concurrent operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the

target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 23, Friske teaches 'allowing a user to select the level of the initial lock' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 24, Friske teaches 'placing an initial lock on the target data includes placing one of the following types of locks on the target data an ACCESS lock; a READ lock; and a WRITE lock' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the

target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13 et seq).

As to claim 25, Friske teaches 'placing a final lock on the target data includes placing an EXCLUSIVE lock on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13 et seq).

As to claim 26, Friske teaches 'placing an initial lock on the target data includes locking an entire table' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would

have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 27, Friske teaches 'receiving the instruction from the user includes receiving an instruction to perform one of the following operations: a CREATE INDEX operation, a COLLECT STASTICS operation, and an ALTER TABLE operation' as after the target data set has been unloaded, the data is ordered in logical sequence in task 410, reorganized in task 412, and loaded into a shadow location in task 414. The target data set may include data indexes which, after the target data set has been reorganized, may be rebuilt into reorganized data indexes in task 416. Rebuilding the data indexes is necessary in the preferred embodiment as discussed above so that quick access to the reorganized data may occur. Log records are applied to the target data set in the shadow location in task 418 which allows any changes to the original data set which occurred while the reorganization was taking place to be applied to the reorganized target data set (coll. 7, lines 55-67).

As to claim 30, Friske discloses a method for use in managing data in a database system (col. 2, 60-67). Friske teaches 'receiving an instruction from a user to perform a

data-definition operation on a set of target data' as the unloaded target data set is reorganized by the processor 106 and loaded into a shadow location 310 of the storage unit 108 (col. 6, lines 25-27 et seq). Friske teaches 'placing an initial lock on the target data at a level that prevents at least one type of concurrent operation on the target data' the non-blocking drain allows the reorganization process to lock and queue while earlier-processes--processes which requested database access before the reorganization process--to complete their routine. At the same time, database access by later-processes, that is, processes requesting database access after the reorganization process, is not impeded by the non-blocking drain (see col. 3, lines 29-36). Further, Friske teaches 'initiating execution of the operation on the target data' as providing substantially continuous access to the database while reorganizing process is executing or waiting to execute. A data set is subject to reorganize in the target data set such as set of pages from the logical database (see col. 3, lines 26-27 and col. 6, lines 5-7). Finally, Friske teaches 'at some point after execution has begun, placing a lock on the target data to prevent concurrent execution of other operations on the target data' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the

target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

Friske does not explicitly indicate claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data.

Cejtin discloses claimed placing a lock on the target data to prevent concurrent execution of other operations on the target data (If the target address space of a remote-run! operation is the current address space, a new thread is simply spawned to execute the application. Otherwise, a message is constructed and sent across the channel connecting the source and target address spaces. The send-message procedure is responsible for writing a message to the target address space. Besides the actual message, send-message requires knowing the type of message being sent; in this case, the type is "run". It also needs to know the appropriate output channel, and requires access to a lock to prevent other messages from being sent to the target by threads concurrently executing on its address space while the transfer is underway. Send-message linearizes the message and associated data and releases the lock when complete, see col. 10, lines 18-31, Cejtin).

It would have been obvious to one ordinary skill in the data processing art at the time of the present invention to combine the teachings of the cited references because placing a lock on the target data to prevent concurrent execution of other operations on the target data of Cejtin teaching would have allowed Friske's system to much progress in building type systems and optimizers for Scheme that catch many potential type

errors statically in order to significantly alleviate debugging overheads that would otherwise be incurred as suggested by Cejtin at col. 5, lines 29-33.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

8. Claims 28 and 29 are rejected under 35 U.S.C. 102(e) as being anticipated by Friske et al. ('Friske' hereinafter), US Patent 6,070,170.

As to claim 28, Friske discloses a method for use in managing data in a database system (col. 2, 60-67). Friske teaches 'receiving a request to perform a MODIFY DATABASE/USER operation on a set of target data' as the unloaded target data set is reorganized by the processor 106 and loaded into a shadow location 310 of the storage unit 108 (col. 6, lines 25-27 et seq). Further, Friske teaches 'initiating execution of the operation' as a program of machine-readable instructions executable by a digital data processing apparatus to perform a method for reorganizing a database (col. 3, lines 20-22). Finally, Friske teaches 'at some point after execution has begun, placing a lock on the target data to prevent concurrent execution of other operations on the target data'

as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13).

As to claim 29, Friske teaches 'maintaining an ACCESS lock on the target database or user and no locks on the immediate parent of the targeted database or user during execution of the MODIFY DATABASE/USER operation' as a blocking drain has been used to request a lock on a target data set for data reorganization purposes. The process requesting the lock, B, would have to wait for the target data set if another process, A, already had a lock on the target data set. If another process, C, came along and requested access to the target data set, it would be placed in a queue behind B waiting for access to the data set. Assuming the reorganization process B and process C were queued and waiting behind A for the target data set, the unload phase shown as task 408 would have to wait to use the target data set until all active logical work units (LUW) of process A were completed, where a LUW includes the processing a program

performs between synchronization ('concurrent') points with the apparatus 100 (col. 7, lines 1-13 et seq).

Remarks

9. **First**, Applicants argue that Friske does not teach, 'executing the operation in the database on the set of target data'.

In response to the applicant's arguments, the Examiner respectfully submits in particular, Friske teaches this limitation as, providing substantially continuous access to the database while reorganizing process is executing or waiting to execute. A data set is subject to reorganize in the target data set such as set of pages from the logical database, see col. 3, lines 26-27 and col. 6, lines 5-7.

Second, Applicants argue that Friske does not teach, 'modify database/user operation on a set of data in a database system'.

In response to the applicant's arguments, the Examiner respectfully submits in particular, Friske teaches this limitation as, the log records update ('modify') the target data to the logical equivalent of the original data set and data set substantially remains in reorganized form, see col. 6, lines 37-39.

Third, Applicants argue that Friske does not teach, 'initial lock and final lock,..'.

In response to the applicant's arguments, the Examiner respectfully submits in particular, Friske teaches this limitation as, the non-blocking drain allows the reorganization process to lock and queue while earlier-processes the processes which requested database access before the reorganization process to complete their routine.

At the same time, database access by later-processes, that is, processes requesting database access after the reorganization process, is not impeded by the non-blocking drain, see col. 3, lines 29-36.

Hence applicant's arguments do not distinguish over the prior art of record.

In light of the forgoing arguments, the 103/102 rejections are hereby sustained.

Contact Information

10. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mohammad Ali whose telephone number is (571) 272-4105. The examiner can normally be reached on Monday-Thursday (7:30 am-6:00 pm).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Hosain T. Alam can be reached on (571) 272-3978. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Mohammad Ali
Primary Examiner
Art Unit 2166

MA
July 8, 2006

HOSAIN ALAM
SUPERVISORY PATENT EXAMINER